Writing Tests for Biopython modules

Brad Chapman (chapmanb@uga.edu)

July 5, 2008

Contents

1 Purpose and Justi cations

So you want to write a Test for a Biopython module? Great! Providing comprehensive tests for modules is one of the most important parts of keeping code up to date and working the way you expect it to. It also tends to be one of the most undervalued aspects of contributing, so this document is designed to make writing good test code as easy as possible.

We start o with the simple assumption that there is a module you wrote (or which doesn't already have tests), and you want to test it out. We'll call it `MyModule`, for lack of a better name, from now on.

2 Understanding the Biopython test system

Biopython tests are found in `biopython/Tests` and each test will have two important les and directories involved with it:

1. `test_MyModule.py` { The actual test code for your module.
2. `MyModule` { A directory where any necessary input les will be located. Any output les that will be

```
import os
import sys
import unittest

def run_tests(argv):
    test_suite = testing_suite()
    runner = unittest.TextTestRunner(sys.stdout, verbosity = 2)
    runner.run(test_suite)

def testing_suite():
    """Generate the suite of tests.
    """
    test_suite = unittest.TestSuite()

    test_loader = unittest.TestLoader()
    test_loader.testMethodPrefix = 't_'
    tests = [MyModuleTestOne]
```

```
"""Test to be sure that MyModule can parse inmcan
"



"En5(sure)-525(that)-52wule can
"
```

executed, and then checked against the expected output (all 'ok's) to make sure nothing has broken. Well, you need to make sure the tests knows what the expected output is.

To do this takes just a second: